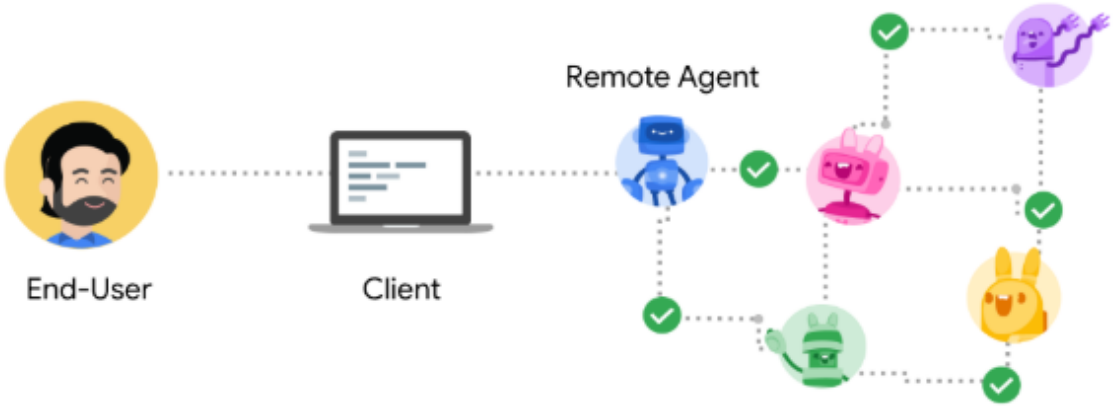
Agent-to-Agent

I. Giới thiệu

1. Khái niệm

Là một tiêu chuẩn mở mới được phát triển bởi Google nhằm mục tiêu giải quyết một thách thức cơ bản trong lĩnh vực AI: làm thế nào để các tác nhân AI (AI agents), được xây dựng bởi các đội ngũ khác nhau, sử dụng các công nghệ khác nhau và thuộc sở hữu của các tổ chức khác nhau, có thể giao tiếp và cộng tác hiệu quả?

AI Agents: Có thể hiểu đơn giản là bao gồm mô hình ngôn ngữ lớn (LLMs) kết hợp với các tools, qua đó nó có thể gọi các tools để thực hiện hành động được chứ không phải chỉ đơn thuần là sinh ra văn bản thông thường.



Ý tưởng chính: Mỗi agent sẽ có một endpoint https (để call đến) và một "danh thiếp", mô tả xem nó đảm nhận chức vụ gì và có thể làm được gì. Sau đó khi công việc đến, ta dùng một Agent điều phối công việc tương ứng với nhiệm vụ của từng agent được ghi trên "danh thiếp". Sẽ được nói rõ hơn ở các phần sau.

Trước khi có A2A, vẫn có khá nhiều cách để các Agent trong hệ thống MAS (Multi-Agents System) chuyển thông tin cho nhau, ví dụ như Handoff, Graph, hay đơn giản chỉ là If-else. Tuy nhiên A2A cung cấp một chuẩn, giúp cho hệ thống mở rộng dễ dàng hơn.

II. Các concepts chính

1. ***Agent Card:*** Như là một danh thiếp cho các Agent, ở đó chứa thông tin Agent đó có thể làm gì, cần gì, bằng một format JSON. Nói cách khác nó kiểu meta data cho mỗi agent.
    - Nó chứa hết mọi thứ như mô tả, I/O, yêu cầu authen, skills của mỗi agent.
    - Cần một endpoint chuẩn và đồng nhất cho danh thiếp của mỗi agent, thường là /well-known/agent.json (theo Google đề xuất). Điều này là để có thể dễ dàng lấy Card của từng Agent được sử dụng.

2. ***A2A Client:*** Cầu nối giữa User và A2A Server, là một AI Agent hoặc ứng dụng khác:
    - Lấy thông tin của các Agent khác từ Agent card.
    - Chuyển yêu cầu thành chuẩn định dạng A2A, sau đấy điều phối cho Remote Agent tương ứng có thể xử lý được yêu cầu, rồi nhận phản hồi.
    - A2A Client có thể sử dụng các cơ chế xác thực (authentication) được mô tả trong Agent Card để đảm bảo giao tiếp an toàn.

3. ***A2A Server:*** Là một AI Agent (ta mặc định trong hệ thống này, các Agent đều phải tuân theo A2A như định dang JSON nó nhận, Agent Card…), được host chạy được, có endpoint https, nhận JSON từ A2A Client, xử lý và response hoặc hỏi thêm câu hỏi phụ nếu cần.

4. ***Concepts về format giao tiếp***

    Là các thành phần của chuỗi JSON được sử dụng để giao tiếp giữa Client và Server, thông qua việc chuẩn hóa chúng, ta có thể dễ dàng kết hợp các hệ thống Agent từ nhiều bên, miễn là chúng theo chuẩn A2A.

    a. ***A2A Task:*** Là đơn vị công việc chính trong A2A, nhìn chung nó là để Client cho mấy con server biết nhiệm vụ là gì.
    - Client sẽ **khởi tạo một task** để đạt một mục tiêu cụ thể (ví dụ: "tạo báo cáo," "đặt vé máy bay," "trả lời câu hỏi"). Nó sẽ tạo ra một ID riêng cho từng task.
    - Mỗi task có lifetime rõ ràng, nó được thể hiện qua một vài status như submitted, working, input-required, completed, failed.

- Mỗi một task có thể cần nhiều turn trao đổi giữa Client và Server để hoàn thành.

b. ***A2A Message:*** Là một turn giao tiếp trong quá trình thực hiện 1 task. Tức là mỗi task có thể có nhiều message giữa client và server.
- Cấu trúc: role: user/agent là client gửi hoặc server trả lời,
- Parts: danh sách các phần nội dung. Mỗi parts sẽ bao gồm type và nội dung bên trong nó, sẽ được ví dụ ở bên dưới. Đây cũng là đơn vị nhỏ nhất, một Message cũng có thể có nhiều Parts.

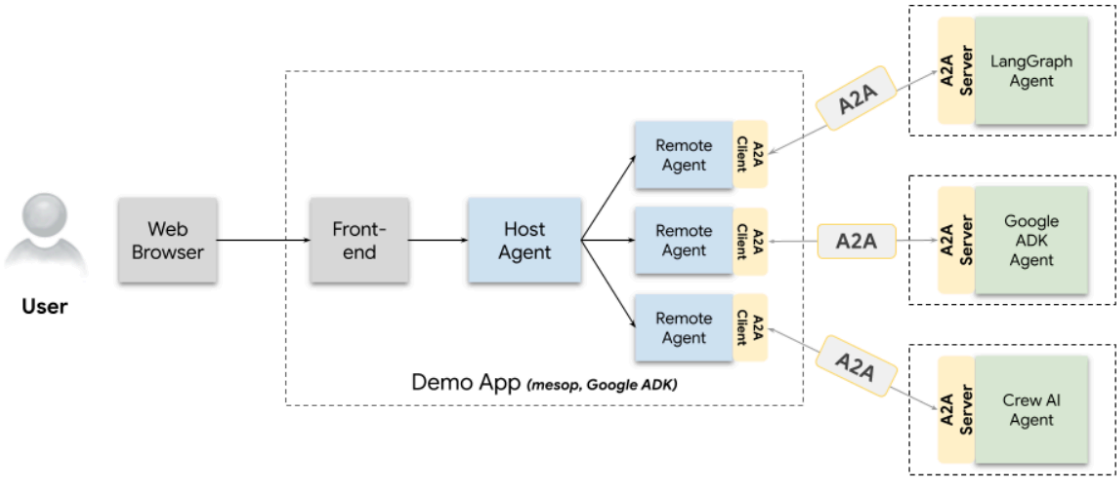
c. ***A2A Artifact:*** Là kết quả đầu ra Server xử lý xong sau đó đưa cho A2A Client, bên trong nó cũng có thể có nhiều Parts.

III. Use-case

A2A không chỉ đơn thuần là một tools, nó dường như là một hạ tầng, thuận tiện cho việc trao đổi, tương tác giữa các AI Agent trong hệ thống Multi-Agents, nên use-case của nó rất rộng.

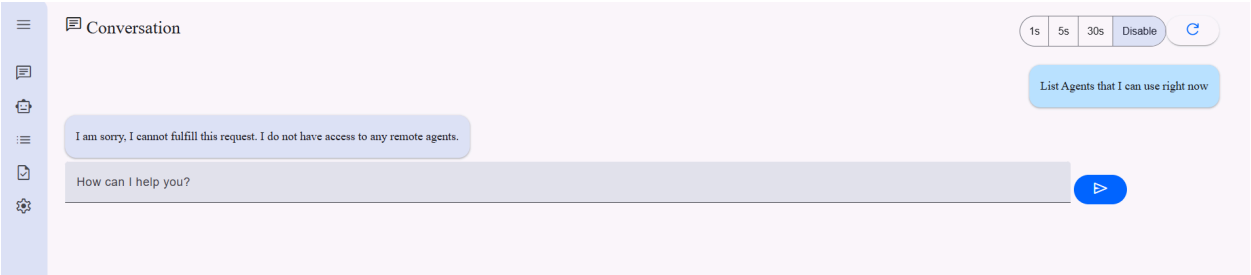
Phổ biến nhất: Agent-to-Agent Communication: Các AI Agents trao đổi thông tin với nhau (đây là use case cơ bản nhất)
- Ví dụ: một agent đại lý du lịch có thể phối hợp với các agent khác như agent hãng hàng không, agent khách sạn và agent thuê xe để đặt toàn bộ một chuyến đi, chỉ cần người dùng đưa ra yêu cầu, giá cả phù hợp trong prompt.

Use-case trong Demo của Google trên repo Google/A2A: Họ build một hệ thống Multi-Agents nhằm nhiệm vụ giải quyết một số vấn đề liên quan đến thu ngân (như hoàn tiền, đổi tiền…), mỗi hoạt động đó được một agent đảm nhận thực hiện:
- Google ADK Agent: được build để hoàn tiền
- LangGraph Agent: được build để thực hiện đổi tiền từ đơn vị này sang đơn vị khác.
- Ở phần trung gian, một Host Agent sẽ đảm nhận vai trò như một điều phối viên.



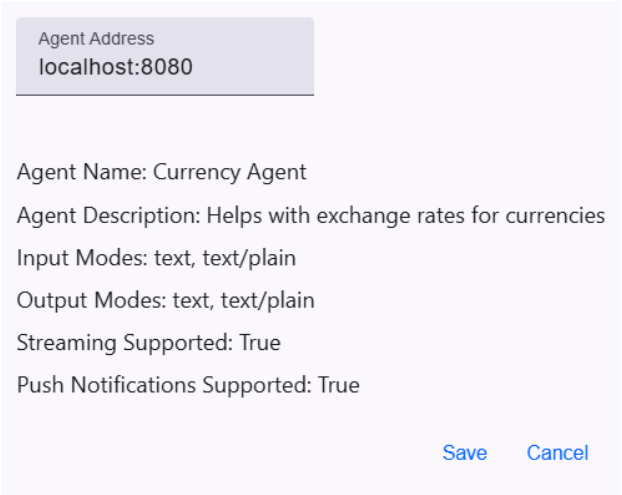
Các Agent phụ có thể được thêm vào thông qua giao diện (ta có thể tùy ý thêm vào bất cứ Agent nào, miễn là nó theo chuẩn A2A), khi ta thêm một Agent mới vào list thông qua cách nhập địa chỉ, A2A Client sẽ sử dụng phương thức GET tới endpoint /well-known/agent.json (ví dụ host Google ADK Agent ở cổng 12002, thì ta có thể lấy được Agent Card của Agent này bằng cách gửi phương thức GET đến địa chỉ localhost:12002/well-known/agent.json

Ví dụ giao diện chat gốc: khi chưa thêm Agent vào và hỏi "List Agents that I can use", hệ thống sẽ không thể tìm được card của các Agents nên sẽ trả ra không thấy:

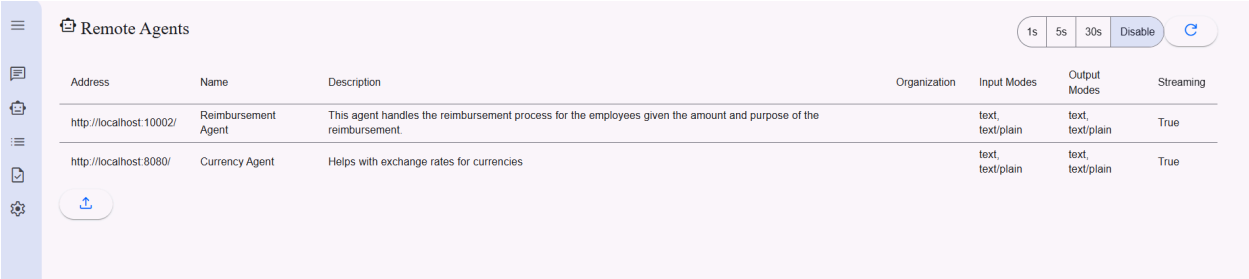


Tiếp theo, ta chuyển qua tab Remote Agents và thêm endpoint của các Agent vào (hiện tại đang tự host tại máy nên địa chỉ sẽ là localhost). Khi các endpoint được thêm vào, phương thức GET được gửi tới địa chỉ localhost:port/well-known/agent.json để lấy ra Agent Card.

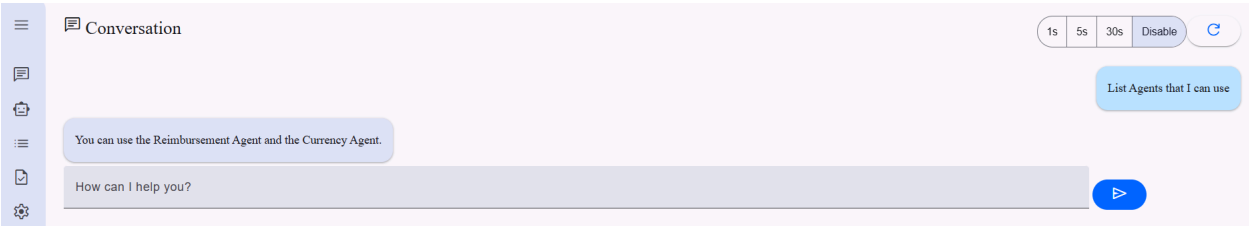
```
INFO:     Started server process [20092]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://localhost:8080 (Press CTRL+C to quit)
INFO:     ::1:53314 - "GET /.well-known/agent.json HTTP/1.1" 200 OK
INFO:     ::1:53325 - "GET /.well-known/agent.json HTTP/1.1" 200 OK
```



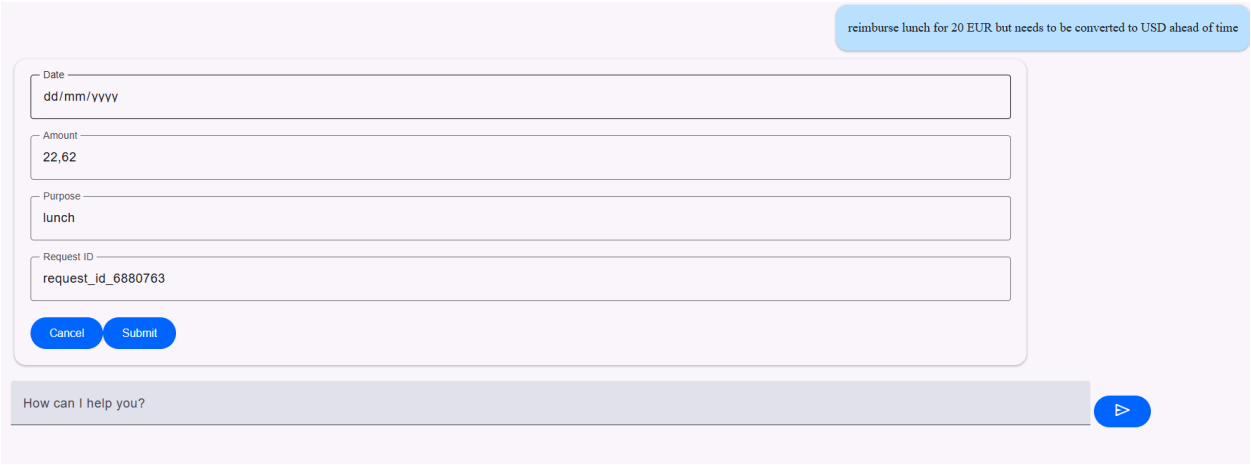
Đây là địa chỉ của LangGraph Agent (đang host ở cổng 8080), theo mô tả trong agent card, nó có thể đổi các đơn vị tiền tệ. Chúng ta sẽ thêm 1 agent nữa là Agent hoàn tiền ở cổng 10002. Như vậy ta có 2 Agents đã được thêm vào list, các mô tả ngắn gọn về I/O, endpoint, việc có thể thực hiện được hiển thị như bên dưới.



Lúc này khi hỏi list agents, bot đã có thể trả lời các Agents nó có thể sử dụng:



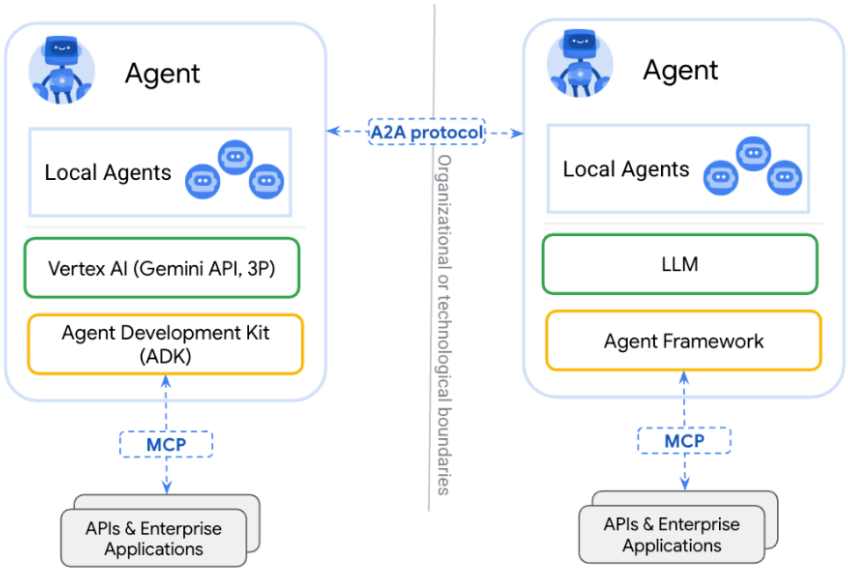
Ta có thể thử hỏi một câu hỏi yêu cầu sự kết hợp của cả 2 Agent mới trả lời được, Host Agent sẽ điều phối đúng cách (thậm chí hỏi lại thông tin) để có thể hoạt động chính xác theo yêu cầu: "*reimburse lunch for 20 EUR but needs to be converted to USD ahead of time" -> cần phải đổi tiền trước rồi tạo hóa đơn hoàn tiền.*



Như vậy, 20 EUR đã được chuyển sang 22.62 Dollar trước khi thực hiện việc hoàn tiền. Host Agent đã điều phối để mọi thứ hoạt động một cách hợp lý.

Về mặt giao diện, người dùng thật sự không cần biết 2 Agent đó được code như nào, miễn là nó tuân theo A2A, ta chỉ cần thêm vào list agent và sau đó hỏi đáp. Host Agent sẽ điều phối câu hỏi đến Agent phù hợp, dựa trên Agent Card của từng con. Điều này mang đến một khả năng mở rộng rất kinh khủng trong tương lai: có thể phát triển tất cả Agent thành một Agent Market (chợ Agent), sau đó người dùng chỉ việc lựa chọn Agent dựa trên Agent Card của chúng và thêm vào danh sách agent cá nhân, là hoàn toàn có thể sử dụng được một cách dễ dàng.

Agent-to-agent cũng hoàn toàn có thể kết hợp với MCP để tạo thành một hệ thống hoàn chỉnh:



Chúng hoàn toàn có thể kết hợp là vì MCP tổ chức lại cách Agent tương tác với các tools, APIs, công cụ bên ngoài, trong khi đó A2A tổ chức lại cách các Agent giao tiếp với nhau.

IV. Thách thức hiện tại

- A2A là một công nghệ còn rất mới, ngay bản thân của Google cũng đang còn rất nhiều bug, họ đang release liên tục để fix các bug còn tiềm ẩn.
- Chưa được tích hợp thành các thư viện Python -> khó khăn hơn trong cài đặt và làm việc.
- Yêu cầu phiên bản Python rất cao (3.12 ++), điều này có thể gây ảnh hưởng đến các hệ thống cũ nếu sử dụng các công nghệ, thư viện đã lâu chưa update.

Nhưng đổi lại, nó cũng có nhiều ưu điểm, đặc biệt là tiềm năng mở rộng:

- Mã nguồn mở với Apache Lincense 2.0. Hòan toàn có thể sử dụng với mục đích thương mại.
- Khả năng mở rộng lớn.
- Nhà phát triển lớn: Google cùng các đối tác dường như muốn tạo ra một hệ sinh thái nhờ vào A2A.

V. Tài liệu tham khảo

Github Google A2A: https://github.com/google/A2A
Docs Google A2A: https://google.github.io/A2A/